МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное автономное образовательное учреждение высшего образования

«САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ   
АЭРОКОСМИЧЕСКОГО ПРИБОРОСТРОЕНИЯ»

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| К ЗАЩИТЕ | |  | | | | |  |
| РУКОВОДИТЕЛЬ  доктор педагогических наук доцент | |  |  |  | А.Г. Степанов | | |
| должность, уч. степень, звание | |  | подпись, дата |  | инициалы, фамилия | | |
|  | |  | | | | |  |

ЗАДАНИЕ

НА ВЫПОЛНЕНИЕ КУРСОВОЙ РАБОТЫ ПО ДИСЦИПЛИНЕ ТЕХНОЛОГИИ ПРОГРАММИРОВАНИЯ

ТЕМА КУРСОВОЙ РАБОТЫ:

РАЗРАБОТКА ПРОГРАМНОГО ПРОДУКТА, ПРЕДСТАВЛЯЮЩЕГО ИЗ СЕБЯ КОНСОЛЬНОЕ ПРИЛОЖЕНИЕ, ПОЗВОЛЯЮЩЕЕ ПЕРЕВЕСТИ ЦЕЛОЕ ПОЛОЖИТЕЛЬНОЕ ЧИСЛО В СИСТЕМЫ СЧИСЛЕНИЯ (ДВОИЧНАЯ, ВОСЬМИРИЧНАЯ, ШЕСТНАДЦАТИРИЧНАЯ, ФАКТОРИАЛЬНАЯ, ФИБОНАЧЧИЕВА)

|  |  |  |
| --- | --- | --- |
| Задание выдано студенту группы № 8026 |  | Воронину И.Н. |
|  |  |  |

Санкт-Петербург

2022

Требования к оформлению пояснительной записки

Пояснительная записка оформляется как единый текстовый документ, содержащий разделы, подразделы и, при необходимости, пункты и подпункты. Она должна начинаться с титульного листа и задания на курсовую работу, утвержденного руководителем. Пояснительная записка оформляется в соответствии с требованиями стандартов «Правила оформления текстовых документов» ГОСТ 7.32-2017 издания 2017 года. https://guap.ru/standart/doc). Там же приведены примеры оформления библиографических ссылок.

Календарный план выполнения работы и проведения консультаций

|  |  |  |  |
| --- | --- | --- | --- |
| Номер недели в семестре | | Основная тема консультации | Основное содержание работы, выполняемой студентом между консультациями |
| осень | весна |
| 1-2 | 23-24 | Вводное занятие, составление списка группы и закрепление студентов за конкретными преподавателями. Разъяснение порядка выбора предметных тем курсовой работы | Анализ литературы по теме работы. Выбор и формулировка предметной темы работы. |
| 3-4 | 25-26 | Обсуждение предложенных студентами тем курсовых работ | Подготовка и заполнение бланка задания на выполнение работы. Отправка его преподавателю и получение подтверждения об утверждении задания. |
| 5-6 | 27-28 | Утверждение заданий на выполнение курсовой работы | Разработка Технического задания на разработку программного изделия. Отправка его преподавателю и получение подтверждения о его утверждении. |
| 7-8 | 29-30 | Утверждение Технического заданий на разработку программного изделия | Декомпозиция задачи и разработка общего алгоритма программного изделия. Отправка разработанного алгоритма преподавателю и получение подтверждения о его утверждении. |
| 9-10 | 31-32 | Разработка тестов. Программирование. | Разработка тестов. Программирование задачи (диалоги, графика, файловая работа, математические вычисления). Отправка результатов программирования преподавателю и получение подтверждения об их утверждении. |
| 11-12 | 33-34 | Обсуждение результатов решения задачи и ее тестирования. | Подготовка пояснительной записки, отправка ее преподавателю на предмет ее утверждения. Подготовка структуры доклада. Подготовка перечня слайдов для выступления |
| 13-14 | 35-36 | Утверждение пояснительной записки. Обсуждение структуры доклада. Обсуждение перечня слайдов. | Подготовка окончательного варианта текста доклада. Подготовка публичного выступления |
| 15-16 | 37-38 | Публичная защита курсовой работы. | Запись на защиту. Защита. |

|  |  |  |
| --- | --- | --- |
|  |  |  |
| Задание принял к исполнению |  | (Воронин И.Н.) |
|  |  |  |
| Задание выдал |  | (Степанов А. Г.) |

СОДЕРЖАНИЕ

[ВВЕДЕНИЕ 4](#_Toc127276206)

[1 Техническое задание 5](#_Toc127276207)

[1.1 Основания для разработки 5](#_Toc127276208)

[1.2 Назначение разработки 6](#_Toc127276209)

[1.3 Требования к программе или программному изделию 6](#_Toc127276210)

[1.4 Требования к программной документации 7](#_Toc127276211)

[1.5 Технико-экономические показатели 8](#_Toc127276212)

[1.6 Стадии и этапы разработки 8](#_Toc127276213)

[1.7 Порядок контроля и приёмки 8](#_Toc127276214)

[1.8 Приложения 9](#_Toc127276215)

[2 Текст программы 10](#_Toc127276216)

[3 Описание программы 14](#_Toc127276217)

[3.1 Общие сведения 14](#_Toc127276218)

[3.2 Язык программирования 14](#_Toc127276219)

[3.3 Функциональное назначение 14](#_Toc127276220)

[3.4 Описание логической структуры 14](#_Toc127276221)

[4 Описание применения 21](#_Toc127276222)

[4.1 Назначение программы 21](#_Toc127276223)

[4.2 Основная область применения 21](#_Toc127276224)

[4.3 Условия применения 21](#_Toc127276225)

[4.4 Описание задачи 21](#_Toc127276226)

[5 Программа и методика испытаний 22](#_Toc127276227)

[ЗАКЛЮЧЕНИЕ 26](#_Toc127276228)

[СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ 27](#_Toc127276229)

## ВВЕДЕНИЕ

Актуальность темы заключается в упрощении процесса перевода целого положительного числа в такие системы счисления как: двоичная, восьмиричная, шестнадцатиричная, факториальная и фибоначчиева. Ручной перевод в данные системы счисления занимает достаточно большое количество времени, поэтому создание такого консольного приложения будет полезно для ускорения и облегчения этого процесса.

Цель работы: выработка навыка проектирования программного продукта способного выполнять поставленную задачу.

Предмет работы: разработка программного продукта, представляющего из себя консольное приложение, позволяющее перевести целое положительное число в различные системы счисления.

Объект работы: учебный процесс.

## 1 Техническое задание

Необходимо разработать программный продукт, представляющий из себя консольное приложение, позволяющее перевести целое положительное число в такие системы счисления, как:

1. Двоичная;
2. Восьмеричная;
3. Шестнадцатеричная;
4. Факториальная;
5. Фибоначчиева.

## 1.1 Основания для разработки

Основание для разработки - учебный план кафедры 82 на 5-й семестр, для обучающихся по направлению 09.03.03(прикладная информатика).

Методического пособие при разработке- «Объектно-ориентированный анализ и программирование».

Основная теоретическая часть данной курсовой работы была взята из источников, представленных в LMS, а именно:

1. Справочник по специальным функциям с формулами, графиками и математическими таблицами: пер. с англ. / под ред. М. Абрамовица, И. Стиган. М.: Наука, 1979, p. 830.
2. Корн Г., Корн Т. Справочник по математике для научных работников и инженеров. М.: Наука, 1970, p. 720.
3. Бронштейн И. Н., Семендяев К. А. Справочник по математике для инженеров и учащихся втузов. М.: Наука, 1986, p. 544

Организация, утвердившая этот документ: ГУАП.

Тема: Разработка программного продукта, представляющего из себя консольное приложение, позволяющее перевести целое положительное число в такие системы счисления, как:

1. Двоичная;
2. Восьмеричная;
3. Шестнадцатеричная;
4. Факториальная;
5. Фибоначчиева.

## 1.2 Назначение разработки

Программный продукт должен использоваться для перевода заданного числа в одну из перечисленных систем счисления.

## 1.3 Требования к программе или программному изделию

1.3.1 Требования к функциональным характеристикам;

Необходимо создать консольное приложение, «переводящее» целое положительное число в запрашиваемую систему счисления, сопровождающее ввод и вывод данных соответствующими сообщениями, по типу «Введите число» и т.д. При невозможности перевода числа необходимо вывести сообщение об ошибке.

Число для перевода должно извлекаться из .xls-файла, либо вводиться через командную строку.

Число, переведенное в системы счисления, необходимо вывести в консольное приложение, либо в соответствующую ячейку в .xls – файле.

Диалог с пользователем необходимо реализовать в консоли.

1.3.2 Требования к надёжности;

Для обеспечения надежного функционирования программы следует обратить внимание на вводимое число (число не должно быть дробным, отрицательным).

1.3.3 Условия эксплуатации;

Условием работы программы является наличие персонального компьютера с предустановленной операционной системой Windows и интегрированную среду разработки PyCharm версии 2020.3.3, а также предустановленным Python 22.0.4

1.3.4 Требования к составу и параметрам технических средств;

Для работы программы компьютер должен иметь рекомендуемую конфигурацию:

* CPU: 2.40GHz (2 ядра),
* RAM: 512 Mb;
* HDD: 100 Gb;
* клавиатура;
* манипулятор “мышь”;
* монитор.

1.3.5 Требования к информационной и программной совместимости;

Программный продукт должен быть написана на языке Python

1.3. 6 Требования к маркировке и упаковке

Репозиторий с главным файлом «main» и со всеми сопутствующими файлами.

1.3.7 Требования к транспортированию и хранению

Созданный файл может быть транспортирован любым возможным способом, исключая ненадежные пути передачи данных. Исходный код программного продукта должен храниться в частном репозитории специального аккаунта на платформе github.

* + 1. Специальные требования

Нет.

## 1.4 Требования к программной документации

Программная документация должна включать в себя:

1. «Техническое задание на разработку программного изделия».
2. «Текст программы».
3. «Описание программы».
4. «Описание применения».
5. «Программа и методика испытаний».

Раздел «Текст программы» представляет собой самостоятельный документ в составе пояснительной записки и содержит результаты программирования. Содержимое раздела оформляется в соответствии с ГОСТ 19.401-78.

Раздел «Описание программы» представляет собой самостоятельный документ в составе пояснительной записки и оформляется в соответствии с требованиями ГОСТ 19.402-78. Описание программы должно содержать следующие подразделы: общие сведения, функциональное назначение, описание логической структуры, используемые технические средства, вызов и загрузка, входные данные, выходные данные.

Раздел «Описание применения» представляет собой самостоятельный документ в составе пояснительной записки и оформляется в соответствии с требованиями ГОСТ 19.502-78. Описание применения должно содержать следующие подразделы: назначение программы, условия применения, описание задачи, входные и выходные данные.

Раздел «Программа и методика испытаний» представляет собой самостоятельный документ в составе пояснительной записки и оформляется в соответствии с требованиями ГОСТ 19.301-79. Программа и методика испытаний должна содержать следующие подразделы: объект испытаний, цель испытаний, требования к программе, требования к программной документации, состав и порядок испытаний, методы испытаний.

## 1.5 Технико-экономические показатели

Эконмические эффективность и преимущества невелики.

## 1.6 Стадии и этапы разработки

1. Разработка общего алгоритма

2. Программирование и реализация программы

3. Тесты программы.

4. Оформление пояснительной записки

5. Представление программы.

## 1.7 Порядок контроля и приёмки

Программа должна соответствовать изначально выбранным выше критериям и корректно функционировать. Для проведения приемочных испытаний исполнитель уведомляет заказчика о готовности программного продукта. В течение 5 рабочих календарных дней заказчик формирует приемочную комиссию. Приемочные испытания проведется в соответствии с программой и методикой испытаний. Программа и методика испытаний разрабатывается исполнителем, согласовывается заказчиком. По результатам приемочных испытаний формируется протокол проведения приемочных испытаний. В случае выявления недостатков программного продукта, они вносятся в протокол проведения приемочных испытаний с указанием срока устранения. Протокол проведения приемочных испытаний подписывается обеими сторонами. После устранения выявленных недостатков, приемочные испытания проводятся повторно.

## 1.8 Приложения

Приложения отсутствуют

## 2 Текст программы

from math import sqrt  
  
def main():  
 x = input(**'Введите целое положительное число: '**)  
 if x[0] == **'q'**:  
 return 0  
 if x[0] == **'-'**:  
 print(**'Число не положительное**\n**Попробуйте снова'**)  
 main()  
 return **''** try:  
 check = int(x)  
 except ValueError:  
 print(**'Введенное значение не является целым числом**\n**Попробуйте еще раз'**)  
 main()  
 return **''** choise = int(input(**' 1: Двоичная'  
 '**\n **2: Восьмеричная'  
 '**\n **3: Шестнадцатиричная'  
 '**\n **4: Фактириальная'  
 '**\n **5: Фиббоначиева**\n**В какую систему перевести? '**))  
 print(**'Ответ: '**, x, **' = '**, end=**''**, sep=**''**)  
 x = int(x)  
 if x == 0:  
 return 0  
 unit\_to\_miltiplier = {1: binary(x), 2: octal(x), 3: hex(x), 4: factorial(x), 5: fib(x)}  
 try:  
 return unit\_to\_miltiplier[choise]  
 except x == 0:  
 return 0  
  
  
def binary(x):  
 answerReversed = []  
 answer = 0  
 while x:  
 answerReversed.append(x % 2)  
 x //= 2  
 answerList = list(enumerate(answerReversed))  
 for l in answerList:  
 answer += l[1] \* (10 \*\* l[0])  
 return answer  
  
  
def decodeBin(x):  
 answer = 0  
 i = 2  
 arr = 0  
 while x:  
 if x % 10:  
 answer += i \*\* arr  
 arr += 1  
 x //= 10  
 return answer  
  
  
def octal(x):  
 answerReversed = []  
 answer = 0  
 while x:  
 answerReversed.append(x % 8)  
 x //= 8  
 answerList = list(enumerate(answerReversed))  
 for l in answerList:  
 answer += l[1] \* (10 \*\* l[0])  
 return answer  
  
  
def decodeOctal(x):  
 answer = 0  
 i = 8  
 arr = 0  
 while x:  
 if x % 10:  
 answer += x % 10 \* (i \*\* arr)  
 arr += 1  
 x //= 10  
 return answer  
  
  
def hex(x):  
 answerReversed = []  
 hexal = [**'A'**, **'B'**, **'C'**, **'D'**, **'E'**, **'F'**]  
 answer = **''** while x:  
 end = x % 16  
 if end >= 10:  
 answerReversed.append(hexal[end % 10])  
 else:  
 answerReversed.append(end)  
 x //= 16  
 answerList = answerReversed[::-1]  
 for l in answerList:  
 answer += str(l)  
 return answer  
  
  
def decodeHex(x):  
 answer = 0  
 hexal = [**'A'**, **'B'**, **'C'**, **'D'**, **'E'**, **'F'**]  
 i = 16  
 arr = 0  
 while x:  
 last = x[-1]  
 if last.isdigit():  
 last = int(last)  
 if last:  
 answer += last \* (i \*\* arr)  
 else:  
 answer += (10 + hexal.index(last)) \* (i \*\* arr)  
 arr += 1  
 x = x[:-1]  
 return answer  
  
  
def factorial(x):  
 answer = 0  
 dnum = 1  
 i = 2  
 while x:  
 answer += (x % i) \* dnum  
 x //= i  
 i += 1  
 dnum \*= 10  
 return answer  
  
  
def decodeFact(x):  
 answer = 0  
 i = 1  
 sumI = 1  
 while x:  
 last = x % 10  
 if last:  
 answer += last \* sumI  
 i += 1  
 sumI \*= i  
 x //= 10  
 return answer  
  
  
def fib(x):  
 answer = **''** listFibs = []  
 i1 = 1  
 i2 = 1  
 while x >= i1:  
 listFibs.append(i1)  
 i2, i1 = i1, i2 + i1  
 listFibs.reverse()  
 for y in listFibs:  
 if y <= x:  
 x -= y  
 answer += **'1'** else:  
 answer += **'0'** return int(answer)  
  
  
def decodeFib(x):  
 listFib = []  
 i1 = 1  
 i2 = 1  
 while x:  
 if x % 10 == 1:  
 listFib.append(i1)  
 i2, i1 = i1, i2 + i1  
 x //= 10  
 return sum(listFib)  
  
  
  
def test():  
 for i in range(1, 1000):  
 assert decodeBin(binary(i)) == i  
 assert decodeOctal((octal(i))) == i  
 assert decodeHex(hex(i)) == i  
 assert decodeFact(factorial(i)) == i  
 assert decodeFib(fib(i)) == i  
 return **'Все числа переведены правильно.'**print(main())

## 3 Описание программы

## 3.1 Общие сведения

Наименование программы: программный продукт, представляющий из себя консольное приложение, позволяющее перевести целое положительное число в такие системы счисления, как:

1. Двоичная;
2. Восьмеричная;
3. Шестнадцатеричная;
4. Факториальная;
5. Фибоначчиева.

## 3.2 Язык программирования

Python.

## 3.3 Функциональное назначение

Программа переводит целое положительное число в различные системы счисления. Пользователь получает на экран предложение ввести необходимое для кодирования число, затем интерфейс с выбором системы счисления, а затем программа выдает закодированное число.

На выходе: пользователь получает готовое закодированное число, которое соответствует заданному числу.

## 3.4 Описание логической структуры

Общий алгоритм работы разрабатываемого программного продукта представлен на рисунке 1.

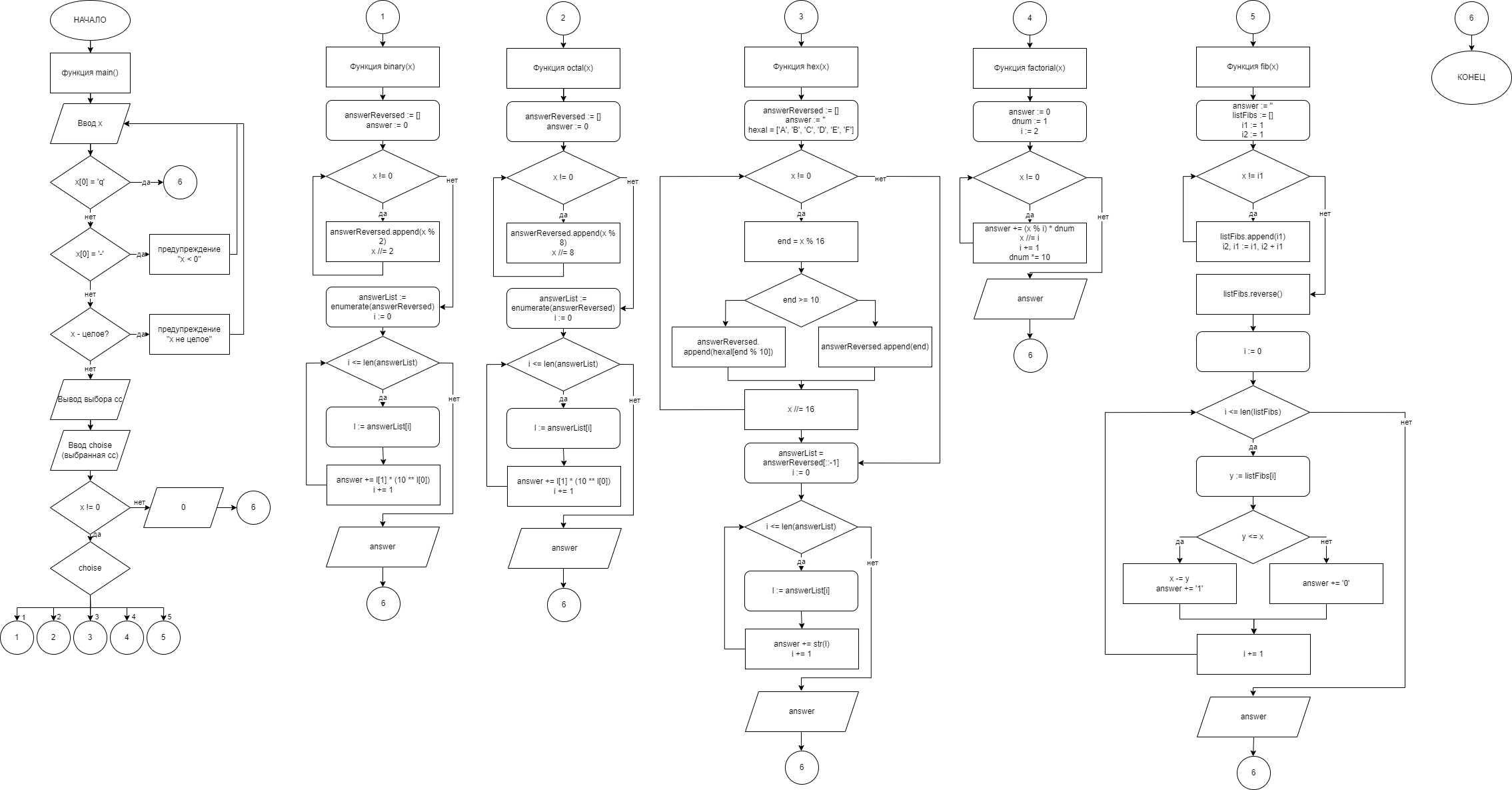


Рисунок 1 – Общий алгоритм работы программы

На рисунке 2 представлен алгоритм перевода числа в двоичную систему счисления.

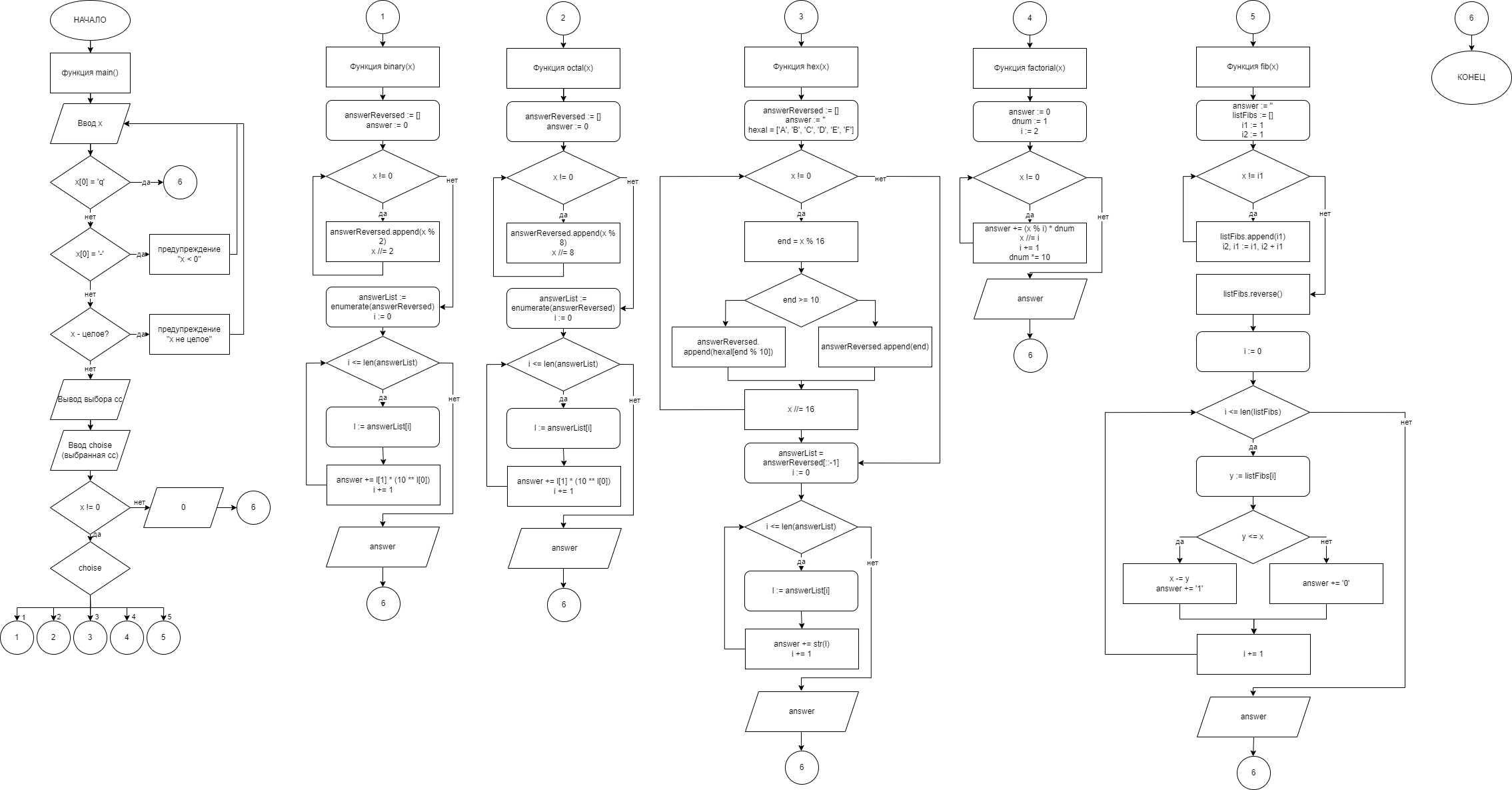


Рисунок 2 – Алгоритм перевода числа в двоичную систему счисления

Алгоритм перевода числа в восьмиричную систему счисления приведен на рисунке 3.

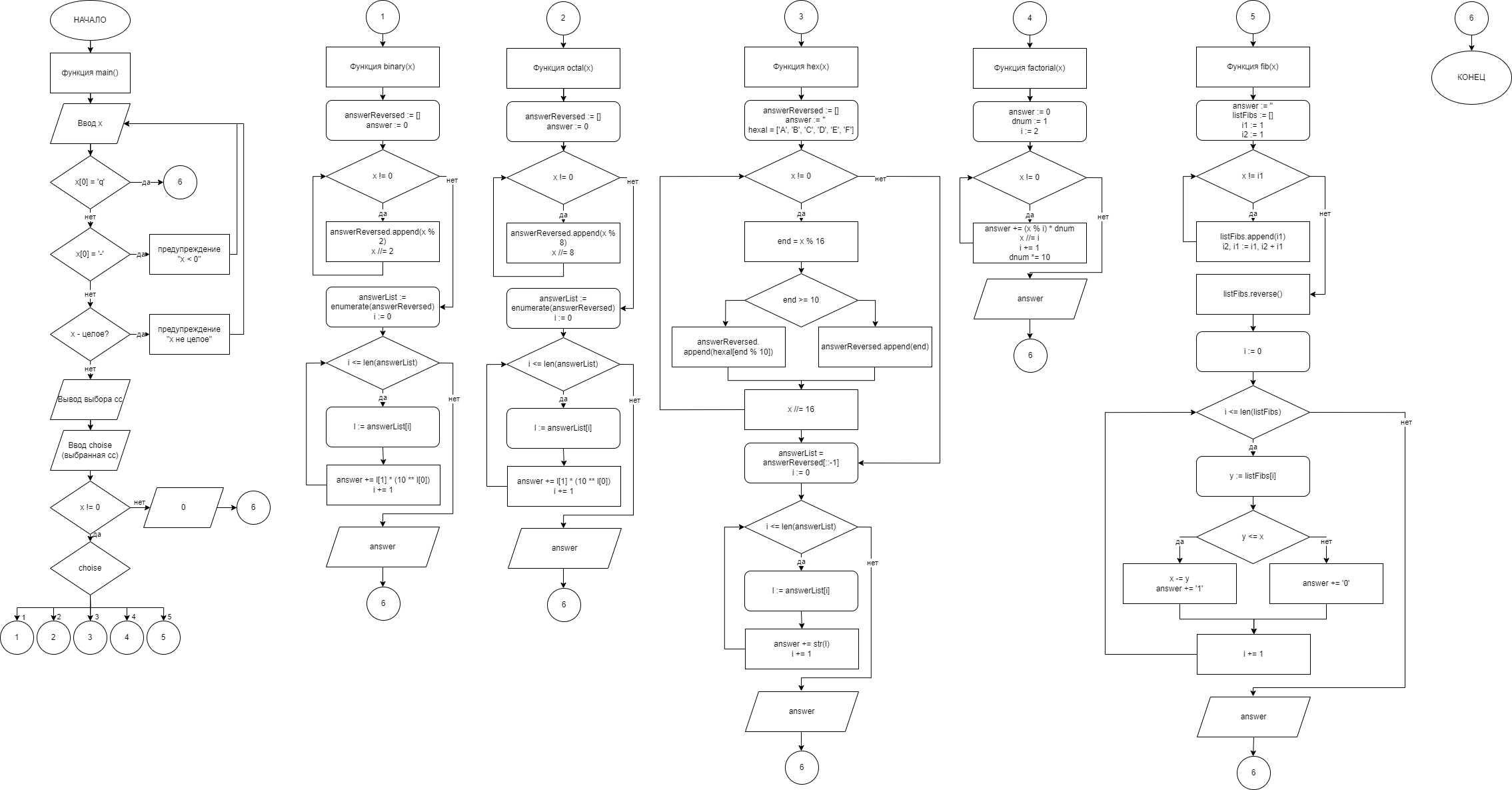


Рисунок 3 – Алгоритм перевода числа в восьмиричную систему счисления

Перевод числа в шестнадцатиричную систему счисления показан на рисунке 4.

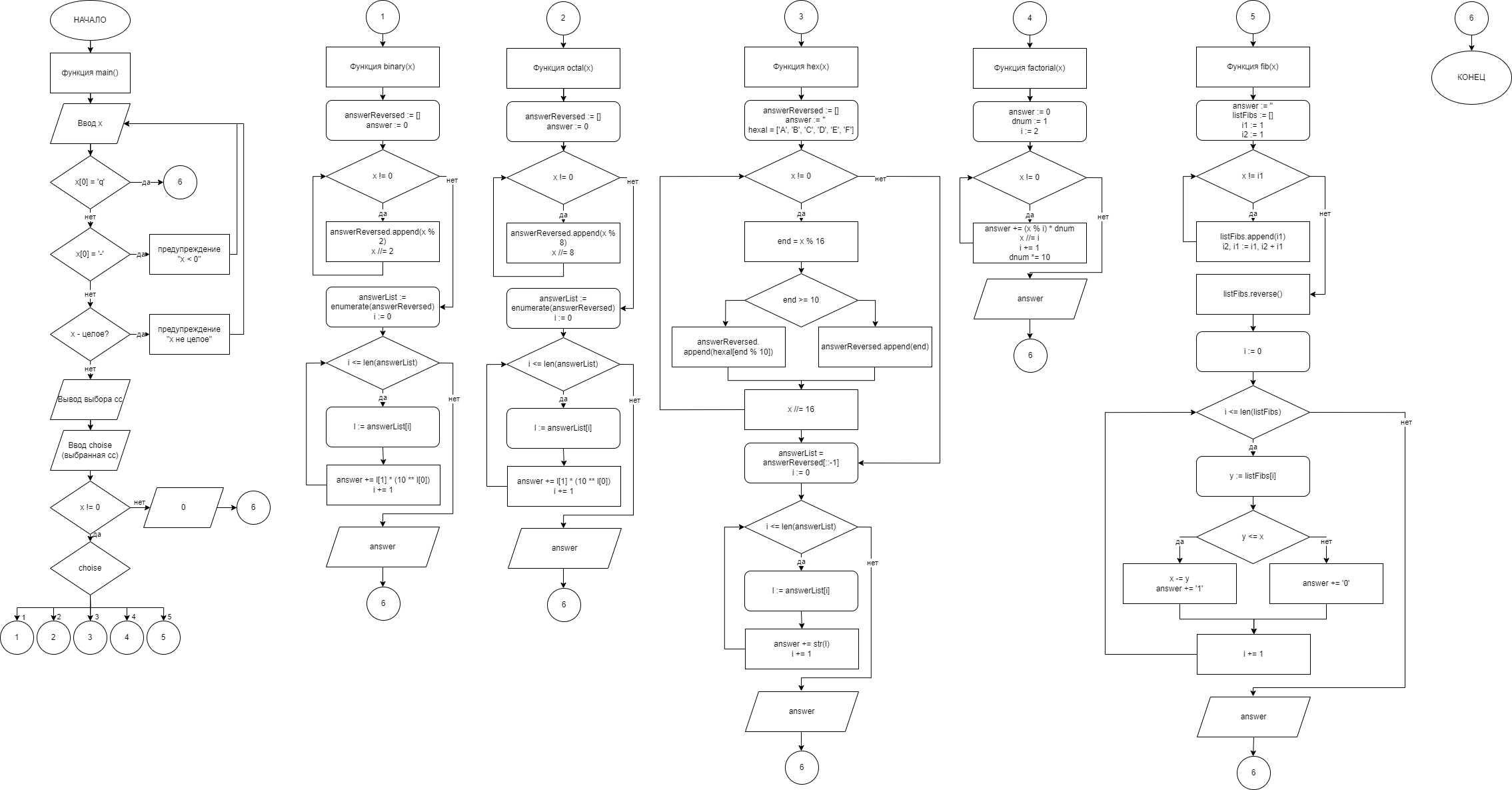


Рисунок 4 – Перевод числа в шестнадцатиричную систему счисления

Рисунок 5 соответствует переводу числа в факториальную систему счисления.

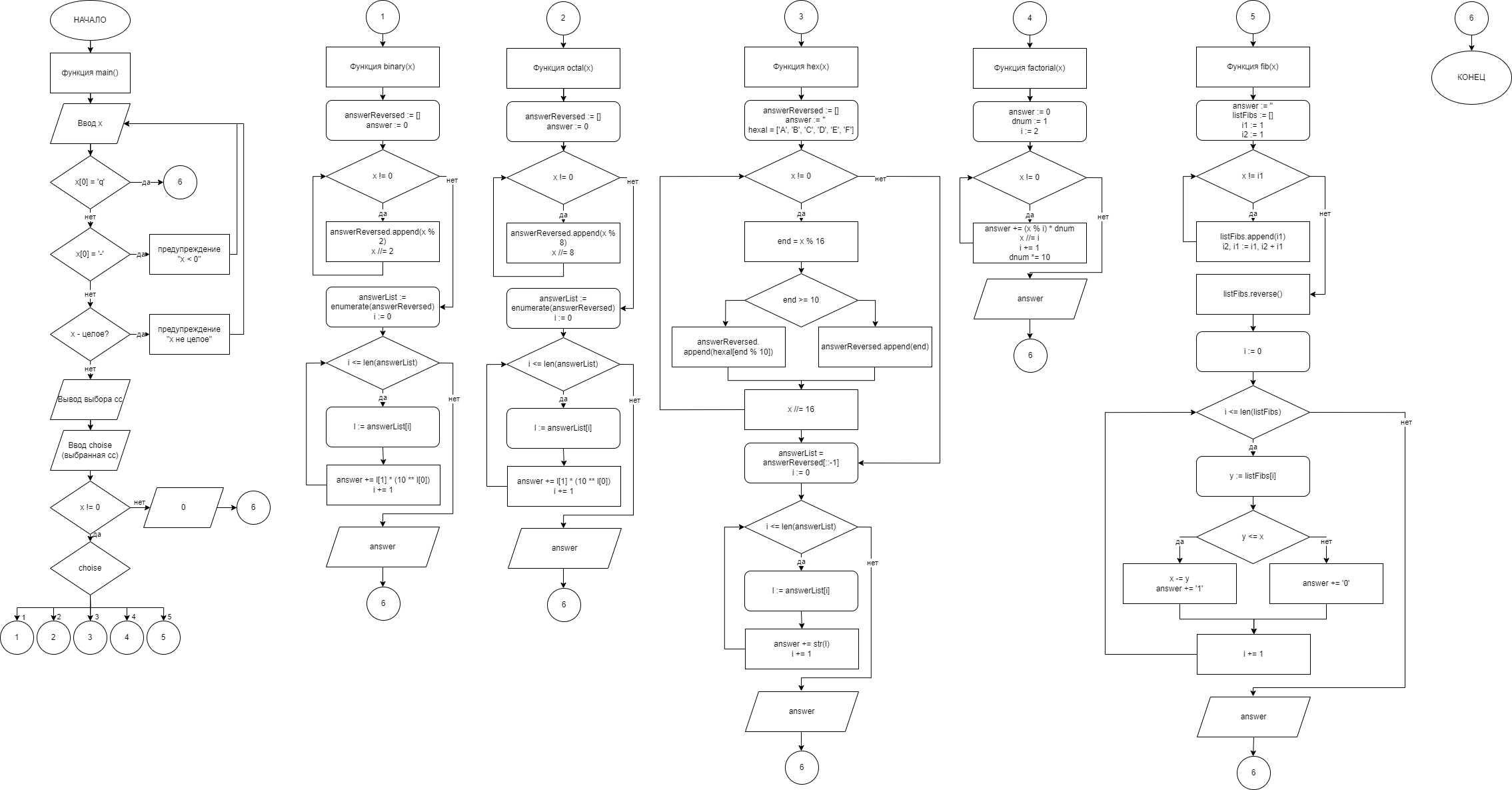


Рисунок 5 – Перевод числа в факториальную систему счисления

Алгоритм перевода числа в фибоначчиеву систему счисления соответствует рисунку 6.

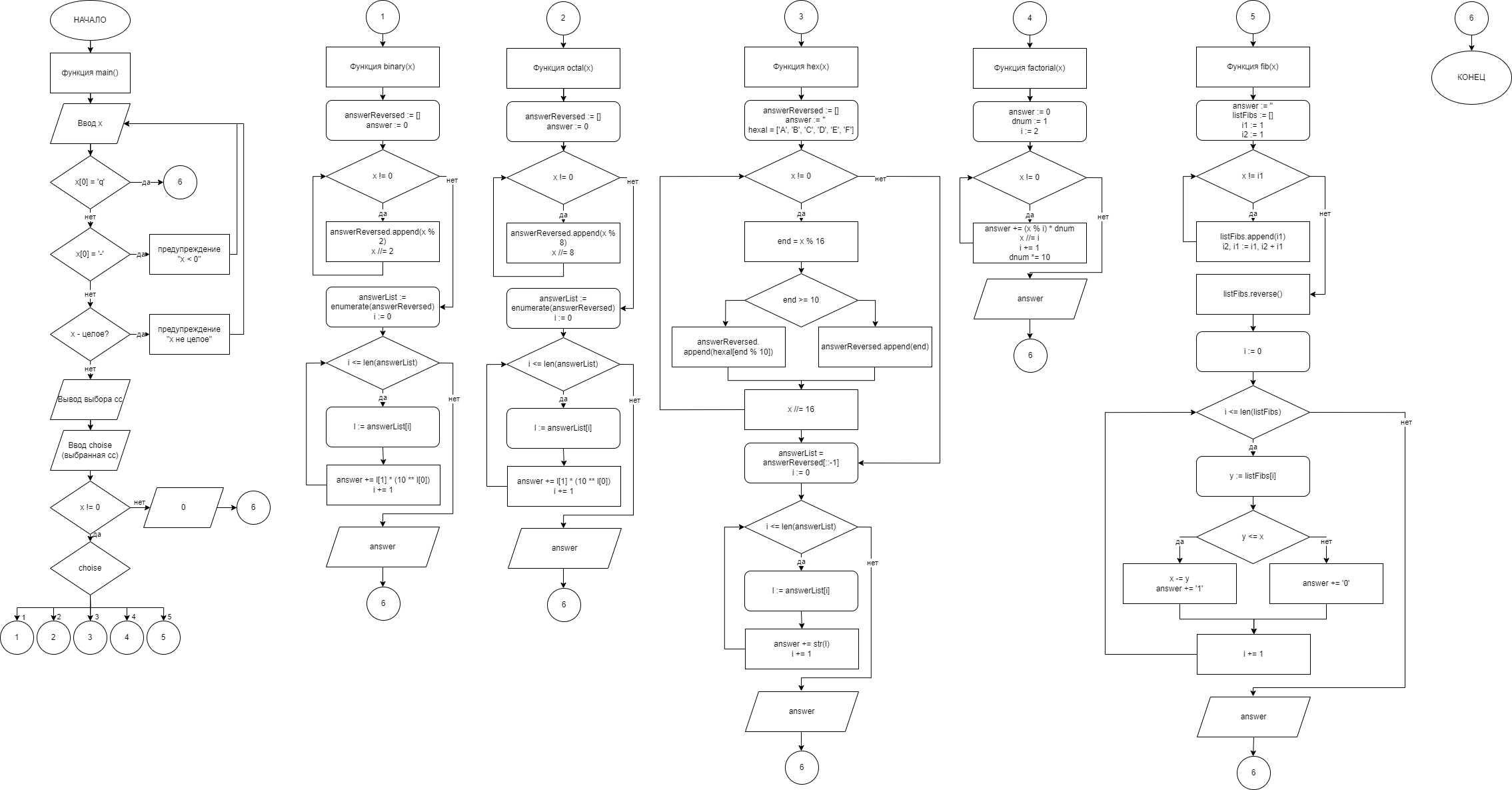


Рисунок 6 – Алгоритм перевода числа в фибоначчиеву систему счисления

## 4 Описание применения

## 4.1 Назначение программы

Программа упрощает и ускоряет процесс перевода целого положительного числа в различные системы счисления. Тем самым дает пользователю преимущество во времени, относительно других. Программа может применятся для решения образовательных вопросов (например сверка конечного результата, полученного путем письменного расчёта, с ответом программы)

## 4.2 Основная область применения

Основной областью применения данной программы можно считать образование (вузовское и довузовское).

## 4.3 Условия применения

Условием применения является наличие персонального компьютера с предустановленной операционной системой Windows и интегрированную среду разработки PyCharm версии 2020.3.3, а также предустановленным Python 22.0.4

## 4.4 Описание задачи

Разработанная программа позволяет пользователю оперативно решать задачи, связанные с переводом целых положительных чисел в различные системы счисления. Благодаря этому вероятность ошибиться у пользователя сводится к минимальной.

## 5 Программа и методика испытаний

Проверка корректности вычислений

Требования к программе:

* точность выполнения программы соответствует ожидаемым результатам;
* программа точно распознает систему счисления, в которую необходимо перевести данное десятичное число.

Методы испытаний.

1. Сравнение данных, полученных в ходе выполнения программы с данными из WolframAlfa и других онлайн калькуляторов (для двоичной, восьмеричной и шестнадцатеричной систем счисления).
2. Самостоятельная разработка функций декодирования к предложенным системам счисления и сравнение исходного числа с декодом, полученного в результате работы программы (для всех систем счисления).

При первом методе проверки программы пользователь получает на экран предложение ввести необходимое для кодирования число, затем интерфейс с выбором системы счисления, а затем программа выдает закодированное число. Пример работы программы представлена на рисунке 7.

Полученное число сверяется с результатом онлайн калькулятора.

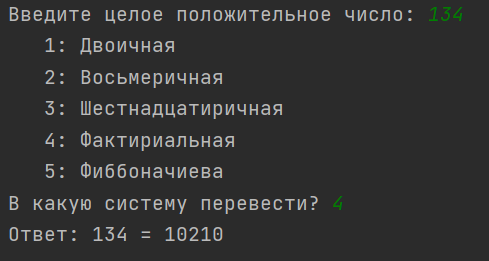


Рисунок 7 – Пример работы программы

Сравнение результатов работы программы и онлайн калькулькулятора для двоичной, восьмиричной и шестнадцатиричной систем счисления приведены в таблице 1.

Таблица 1 – Сравнение результатов

|  |  |  |  |
| --- | --- | --- | --- |
| Число | Двоичная  система счисления | Восьмеричная система счисления | Шестнадцатиричная система счисления |
| *Программа* | *Программа* | *Программа* |
| *WolframAlpha* | *WolframAlpha* | *WolframAlpha* |
| 134 | 10000110 | 206 | 86 |
| 10000110 | 206 | 86 |
| 15 | 1111 | 17 | F |
| 1111 | 17 | F |
| 23 | 10111 | 27 | 17 |
| 10111 | 27 | 17 |
| 590 | 1001001110 | 1116 | 24Е |
| 1001001110 | 1116 | 24Е |
| 1 | 1 | 1 | 1 |
| 1 | 1 | 1 |

Примеры проведения первого метода проверки работы программы представлены на рисунках 8 и 9.

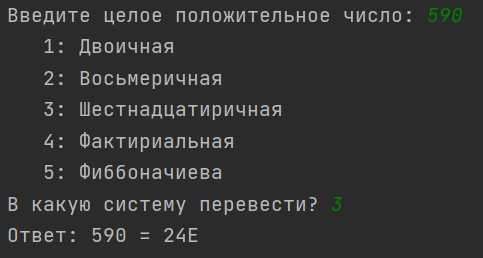


Рисунок 8 – Перевод числа 590 в шестнадцатиричную систему счисления (программа Python)

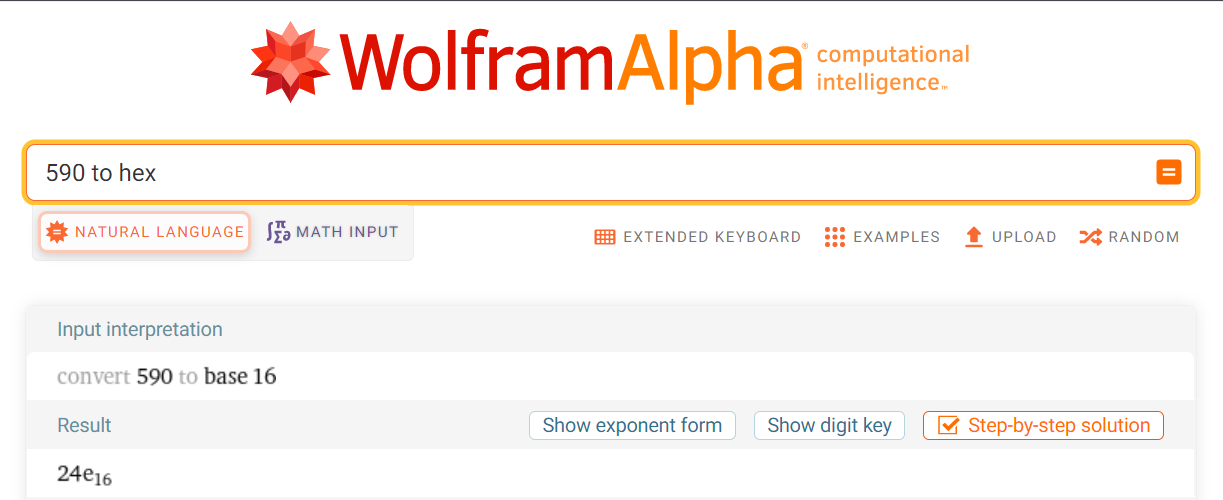


Рисунок 9 – Перевод числа 590 в шестнадцатиричную систему счисления (WolframAlpha)

В ходе выполнения проверки первым методом был проверен интерфейс программы на предмет соответствия вариантов в меню с выполняемыми программой вычислениями. Программа работает исправно, перевод числа в требуемую и выбранную систему счисления производится корректно.

Для реализации второго метода проверки программы на правильность вычислений были прописаны функции декодирования из предложенных систем счисления в десятичную.

Для упрощения проверки данного метода была реализована функция «test», которая перебирает числа от 1 до 1000, кодирует и декодирует их. Результаты сравнивались с исходным числом. Функция «test» вызовет ошибку при первом же отрицательном результате. Код функции представлен на рисунке 10, результат работы функции представлен на рисунке 11.

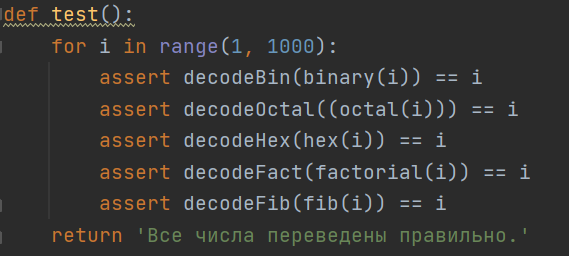


Рисунок 10 – Код функции «test»

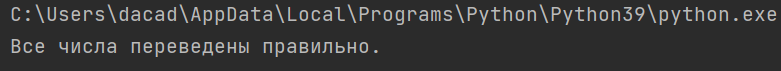


Рисунок 11 – Результат работы функции

## ЗАКЛЮЧЕНИЕ

В ходе тестирования программы было выявлено его соответствие всем основным требованиям, заявленным в техническом задании и методическом пособии «Объектно-ориентированный анализ и программирование».

Разработанная программа отвечает требованиям технического задания. В результате работы мной был укреплён и повышен навык программирования в объектно-ориентированной парадигме и составления алгоритмов.

## СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Нормативная документация». Санкт-Петербургский государственный университет аэрокосмического приборостроения. Available: <http://guap.ru/guap/standart/prav_main.shtml>.
2. «ГОСТ 19.201-78,» Федеральное агентство по техническому регулированию и метрологии РОССТАНДАРТ. Available: http:// protect.gost.ru/v.aspx?control=7&id=155153.
3. «ГОСТ 19.701-90 ЕСПД. Схемы алгоритмов, программ, данных и систем. Условные обозначения и правила выполнения». Available: <http://vsegost.com/Catalog/28/28346.shtml>.
4. Рентабельность предприятия: понятие, виды, показатели оценки // cyberpedia: [Электронный ресурс]. Режим доступа: https://cyberpedia.su/3xfc35.html 9/.
5. NPV - что это, как рассчитать по формуле [Электронный ресурс] // Ранняя пенсия: [сайт]. – <https://retireearly.ru/financial-literacy/npv-chistaja-privedennaja-stoimost>.
6. Марк Дж. Прайс. C# 7 и .NET Core. Кросс-платформенная разработка для профессионалов. 3-е издание // Санкт-Петербург: Издательство «Питер», 2019.